DOCUMENT RESUME

ED 308 237 TM 013 629

AUTHOR Boekkooi-Timminga, Ellen
TITLE The Construction of Parallel Tests from IRT-Based Item Banks. Project Psychometric Aspects of Item Banking No. 43. Research Report 89-2.
INSTITUTION Twente Univ., Enschede (Netherlands). Dept. of Education.
SPONS AGENCY Netherlands Organization for Scientific Research.
PUB DATE Mar 89
NOTE 36p.
AVAILABLE FROM Bibliotheek, Department of Education, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands.
PUB TYPE Reports - Evaluative/Feasibility (142)

EDRS PRICE MF01/PC02 Plus Postage.
DESCRIPTORS Heuristics; *Item Banks; Latent Trait Theory; Mathematical Models; Simulation; *Test Construction; Testing Problems; Test Items
IDENTIFIERS Information Function (Tests); Linear Models; *Parallel Test Forms; Rasch Model; Three Parameter Model; Zero One Programming

ABSTRACT

The construction of parallel tests from item response theory (IRT) based item banks is discussed. Tests are considered parallel whenever their information functions are identical. After the methods for constructing parallel tests are considered, the computational complexity of 0-1 linear programming and the heuristic procedure applied are discussed. Two methods for selecting parallel tests in succession (sequential test construction) are formulated. The first uses a non-partitioned item bank (Method 1), and the second uses a partitioned item bank (Method 2). Two methods are also reviewed for simultaneous test construction, one for non-partitioned item banks (Method 3) and one for partitioned item banks (Method 4). A heuristic procedure is used for solving these 0-1 linear programming problems. A simulation study compared these methods using two item banks, each consisting of 100 items. Satisfactory results were obtained, both in terms of the amount of central processing unit time needed and the differences between the information functions of the parallel tests selected. It was concluded that when the Rasch model fits the items, sequential test construction methods are preferable. For the three-parameter model, the use of Method 1 is inappropriate. Three tables give the results by different methods. (SLD)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED308237

The Construction of Parallel Tests from IRT-Based Item Banks

Research Report 89-2

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

☑ This document has been reproduced as received from the person or organization originating it.

☐ Minor changes have been made to improve reproduction quality.

• Points of view or opinions stated in this document do not necessarily represent official OERI position or policy.

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY

J. NELISSEN

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

Ellen Boekkooi-Timminga

department of EDUCATION

University of Twente

Division of Educational Measurement and Data Analysis

TM013629

Project Psychometric Aspects of Item Banking No.43

Colofon:
Typing: L.A.M. Bosch-Padberg
Cover design: Audiovisuele Sectie TOLAB Toegepaste Onderwijskunde
Printed by: Centrale Reproductie-afdeling

The Construction of Parallel Tests from IRT-Based Item Banks

Ellen Boekkooi-Timminga
Department of Education
University of Twente
Enschede, The Netherlands

To appear in the Journal of Educational Statistics

The construction of parallel tests from IRT-based item banks / Ellen Boekkooi-Timminga - Enschede : University of Twente, Department of Education, March, 1989. - 31 pages

Abstract

The construction of parallel tests from IRT-based item banks is discussed. Tests are considered to be parallel whenever their information functions are identical. Simultaneous and sequential parallel test construction methods based on the use of 0-1 programming are examined. A heuristic procedure is used for solving the 0-1 programming problems. Satisfactory results are obtained, both in terms of the CPU-time needed and differences between the information functions of the parallel tests selected.

Key words: Item Response Theory, Item Banks, Test Construction, Parallel Tests, Information Functions, Linear Programming.

The Construction of Parallel Tests from IRT-based Item Banks

In classical test theory parallel tests play a prominent part, because most reliability coefficient estimation methods assume the existence of parallel tests or forms. Some examples of such estimation methods are alternate-forms methods and internal consistency methods like the Spearman-Brown formula and coefficient alpha. In the latter, existing tests are split into parallel parts, for instance, using the odd-even method or the method of matched random subtests (Gulliksen, 1950). Despite the central role of parallel tests, hardly any algorithms for designing such tests have been developed. One exception is the matched random subtests method, which recently has been algorithmized by van der Linden and Boekkooi-Timminga (1988).

In item response theory very little attention is paid to the construction of parallel tests. This can be explained by the fact that no assumption of parallel measurements has to be made when item response models are used. In educational settings there will be a demand for parallel tests too. In this paper tests are considered to be parallel when their information functions are identical (Samejima, 1977). An exact definition of the concept of information is given by Birnbaum (1968, chapter 17). Here it is assumed that maximum-likelihood estimation is used for the subjects' abilities so that the test information function is the sum of the item information functions.

Furthermore, the test items are assumed to be selected from item banks using 0-1 linear programming from operations research (e.g. Rao, 1985; Salkin, 1975; Taha, 1975; Wagner, 1972). Previous research on modeling test construction problems involving information functions as mathematical programming problems, was already carried out by Adema (1988b), Boekkooi-Timminga (1986, 1987, 1988), Theunissen (1985, 1986), Timminga (1985) and van der Linden and Boekkooi-Timminga (in press). All methods for test construction described in these publications expect the test constructor to specify target test information function values for the test(s) to be constructed at some pre-chosen ability levels.

It seems an obvious procedure to determine parallel tests by sequentially selecting tests from an item bank using the same specifications. However, this approach does not give satisfactory results in terms of the obtained test information functions of the tests (Boekkooi-Timminga, 1986, 1987; Timminga, 1985). In Boekkooi-Timminga (1986, 1987) it was shown that by simultaneously selecting parallel tests fairly identical test information functions could be obtained. However, the major problem of all methods examined so far was the amount of CPU-time needed, which is a feature inherent to 0-1 programming. Simultaneous test construction problems in particular turned out to be very hard, because of the strong increase of the number of decision variables in the model.

In this paper simultaneous and sequential parallel test construction methods are examined. First, the methods for constructing parallel tests are described. Next, the computational complexity of 0–1 linear programming and the heuristic procedure applied in this paper are considered. Finally, the results of a simulation study comparing these methods are outlined.

Four Methods for Constructing Parallel Tests

It is argued that in order to make computerized test construction applicable, at least two conditions must be fulfilled. First, it should be an easy task for the test constructor to specify a target test information function. A procedure that meets this requirement is described by van der Linden and Boekkooi–Timminga (in press). Using this procedure only the shape of the test information function at some well–chosen points at the ability scale needs to be identified. This is much easier than specifying the exact test information function. Van der Linden and Boekkooi–Timminga describe in detail how to elicit these specifications from the test constructor. An approach for deriving an exact target information function is given by Kelderman (1987).

Second, the amount of CPU–time needed to select the tests should be small. The procedure proposed by Adema (1988a), described in the next section, solves 0–1 linear programming problems in a reasonable amount of time. In this paper both the van der Linden and Boekkooi–Timminga (in

press) approach and the Adema procedure are applied in the four methods examined. The 0–1 programming model formulation for each method is given below.

Notation

$i = 1,\ldots,I$	items in the item bank,
$t = 1,\ldots,T$	tests to be constructed,
$k = 1,\ldots,K$	ability levels to be considered,
x_i, x_{it}	decision variables indicating whether an item is (x_i or $x_{it} = 1$) or is not (x_i or $x_{it} = 0$) selected for the test,
r_k	relative amount of target test information at ability level θ_k,
$I_i(\theta_k)$	item information function value of item i at ability level θ_k,
$I_t(\theta_k)$	test information function value of test t at ability level θ_k,
N	number of items to be included in each test,
y	dummy variable.

Sequential Test Construction

Two methods for selecting parallel tests in succession are formulated. The first method uses a non–partitioned item bank and the second a partitioned bank.

Method 1. Model (1) – (5), that is described below, is used for constructing the first test. By maximizing dummy variable y in objective function (1) the total amount of test information obtained is maximized, subject to the constraints

(2) that at least the specified proportions r_k are obtained. The constraint in (3) guarantees that the test consists of N items. The ranges of the variables x_i and y are defined in (4) and (5).

(1) maximize y,

subject to

$$(2) \qquad \sum_{i=1}^{I} I_i(\theta_k)x_i - r_k y \geq 0 \qquad k = 1,\ldots,K$$

$$(3) \qquad \sum_{i=1}^{I} x_i = N$$

$$(4) \qquad x_i \in \{0,1\} \qquad i = 1,\ldots,I$$

$$(5) \qquad y \geq 0.$$

For the other tests to be determined the same model is used. However, the items included in previously constructed tests are excluded from further selection. This can be done by fixing their decision variables to 0 (and thus including some extra constraints), or by removing them from the item bank. The latter possibility is preferred especially when many items have to be excluded, because CPU-time is gained.

The differences between the test information values of the tests can be controlled. Because this difference can be

positive as well as negative two sets of constraints (6) and (7) are added to the model:

$$(6) \qquad \sum_{i=1}^{I} I_i(\theta_k)x_i - (1-p)I_t(\theta_k) \geq 0 \qquad \begin{matrix} k = 1,\ldots,K, \\ t = 1,\ldots,t^*, \end{matrix}$$

$$(7) \qquad \sum_{i=1}^{I} I_i(\theta_k)x_i - (1+p)I_t(\theta_k) \leq 0 \qquad \begin{matrix} k = 1,\ldots,K, \\ t = 1,\ldots,t^*, \end{matrix}$$

where $t = 1,\ldots,t^*$ indicate the tests already constructed; p is the percentage by which the obtained test information values are allowed to differ from the values $I_t(\theta_k)$ of the tests already constructed.

Method 2. This method assumes that the item bank is partitioned into T subsets, comparable in terms of their item information functions, each of the same size $i = 1,\ldots,I/T$; ....., $i = ((t-1)I/T)+1,\ldots,tI/T$; ..... ; $i = ((T-1)I/T)+1,\ldots,I$. From each subset a test is selected, thus, only the indices i in model (1) – (5) have to be adapted. Because each test is selected from a different subset, there is no need for excluding items because of previous usage. Also the constraints in (6) and (7) can be included after a test has been constructed.

It is expected that by partitioning the item bank and subsequently selecting tests from these subsets the amount of CPU-time needed will not be too large.

Simultaneous Test Construction

Two simultaneous test construction methods for parallel test construction are considered. Methods 3 uses a non-partitioned item bank and Method 4 a partitioned bank.

Method 3. The 0-1 programming model is formulated below in (8) - (15). The expressions (8), (9), (10), (14) and (15) are the simultaneous versions of the objective function and constraints in model (1) - (5). By maximizing y the lower bounds to the test information function values will be close to another. The constraints in (11) stipulate that no items are included in more than one test. The maximum difference allowed between the obtained test information values of the tests in percentages of the values of the other tests is constrained in expressions (12) and (13).

(8) maximize y,

subject to

$$(9) \qquad \sum_{i=1}^{I} I_i(\theta_k)x_{it} - r_k y \geq 0 \qquad k = 1,\ldots,K;\ t = 1,\ldots,T$$

$$(10) \qquad \sum_{i=1}^{I} x_{it} = N \qquad t = 1,\ldots T$$

$$(11) \qquad \sum_{t=1}^{T} x_{it} \leq 1 \qquad i = 1,\ldots,I$$

$$(12) \qquad \sum_{i=1}^{I} I_i(\theta_k)x_{it'} - (1-p)\sum_{i=1}^{I} I_i(\theta_k)x_{it} \geq 0 \qquad \begin{array}{l} k = 1,\ldots,K \\ t = 1,\ldots,T-1 \\ t' = t+1,\ldots,T \end{array}$$

$$(13) \qquad \sum_{i=1}^{I} I_i(\theta_k)x_{it'} - (1+p)\sum_{i=1}^{I} I_i(\theta_k)x_{it} \leq 0 \qquad \begin{array}{l} k = 1,\ldots,K \\ t = 1,\ldots,T-1 \\ t' = t+1,\ldots,T \end{array}$$

$$(14) \qquad x_{it} \in \{0,1\} \qquad \begin{array}{l} i = 1,\ldots,I \\ t = 1,\ldots,T \end{array}$$

$$(15) \qquad y \geq 0.$$

An advantage of this model is that when the differences in test information function values are not of interest (leaving out the constraints in (12) and (13)), tests will be selected that have about the same test information values. When a large number of tests have to be constructed, this possibility may be of interest.

Method 4. Like Method 2 it is assumed that the item bank is partitioned into T comparable subsets. From each subset a test is selected simultaneously. The decision variables x_i for $i = 1,..,I/T$; ....., $i = ((t-1)I/T)+1,..,tI/T$; ..... ; $i = ((T-1)I/T)+1,..,I$ denote the items to be included in test 1, 2, ..., T, respectively. The model described for Method 3 can easily be adapted for this case.

(16) maximize y,

subject to

$$(17)\qquad \sum_{i=((t-1)I/T)+1}^{tI/T} I_i(\theta_k)x_i - r_k y \geq 0 \qquad \begin{array}{l} k = 1,\ldots,K \\ t = 1,\ldots,T \end{array}$$

$$(18)\qquad \sum_{i=((t-1)I/T)+1}^{tI/T} x_i = N \qquad t = 1,\ldots,T$$

$$(19)\qquad \sum_{i=((t'-1)I/T)+1}^{t'I/T} I_i(\theta_k)x_i - (1-p) \sum_{i=((t-1)I/T)+1}^{tI/T} I_i(\theta_k)x_i \geq 0 \qquad \begin{array}{l} k = 1,\ldots,K \\ t = 1,\ldots,T-1 \\ t' = t+1,\ldots,T \end{array}$$

$$(20)\qquad \sum_{i=((t'-1)I/T)+1}^{t'I/T} I_i(\theta_k)x_i - (1+p) \sum_{i=((t-1)I/T)+1}^{tI/T} I_i(\theta_k)x_i \leq 0 \qquad \begin{array}{l} k = 1,\ldots,K \\ t = 1,\ldots,T-1 \\ t' = t+1,\ldots,T \end{array}$$

$$(21)\qquad x_i \in \{0,1\} \qquad i = 1,\ldots,I$$

$$(22)\qquad y \geq 0.$$

Note that the constraints in (11) are left out.

Computational Complexity

The parallel test construction problems above are formulated as 0–1 linear programming problems. A 0–1 linear programming problem is a linear programming problem in which the decision variables are restricted to 0–1 values.

Solving a 0–1 linear programming problem optimally involves the following two steps (e.g. Taha, 1975; Williams, 1978): (a) Compute the relaxed 0–1 linear programming problem. The relaxed problem is obtained by dropping the 0–1 constraints on the decision variables; thus, $x_i \in [0,1]$. Doing so, a regular continuous linear programming model is obtained which can be solved quickly, for instance, using the well-known simplex algorithm. The number of fractional decision variable values obtained never exceeds the number of constraints in the model (Dantzig, 1957). (b) Next, the optimal 0–1 solution is determined. Here, a branch-and-bound method is used because these methods have proved to be most successful for integer programming problems. Generally, branch-and-bound methods perform a tree search starting from the relaxed solution. During this search several linear programming problems are solved. These problems are obtained by fixing (bounding) decision variables with fractional values (branching variables) to 0 or 1 in the original relaxed 0–1 linear programming problem. The tree is backtracked when a 0–1 solution is obtained, or when a solution is obtained with an objective function value worse than the best 0–1 solution so far. Then, the last bounded

decision variable is constrained in the other direction. This process of branching, bounding and backtracking is continued until the best 0–1 solution is found. A clear description of this procedure is given in Williams (1978, pp. 146–152).

In comparison to (continuous) linear programming 0–1 linear programming is very complex, because of the large number of linear programming problems that have to be solved. It is commonly conjectured that for 0–1 programming problems no fast algorithms exist (Lenstra & Rinnooy Kan, 1979). Much research has been carried out in this area aiming at approximations. A comprehensive review of this research is given in O'hEigeartaigh, Lenstra and Rinnooy Kan (1985). However, many of the heuristic procedures aim at special types of problems and are not applicable to test construction.

One heuristic approach is to round the fractional decision variables in the relaxed solution. However, no satisfactory results may be obtained, because rounding the fractional decision variable values may result in violating the constraints.

Recently, a quick heuristic procedure especially suitable for test construction problems was proposed by Adema (1988a). A short description of this heuristic, that was used to solve the models proposed in the previous section, is given next. Adema adapts the above described branch–and–bound procedure in two ways:

(a) After the relaxed solution was obtained the algorithm fixes a number of decision variables to 0 or 1 using

their reduced costs. The reduced cost of a decision variable with a value equal to zero or one (nonbasic variable) indicates the amount by which the objective function value z should decrease per unit increase in the variable (Murtagh, 1981; Williams, 1978). Decision variables with values between zero and one (basic variables) have reduced costs of zero. All items with large and small reduced costs are set to 0 and 1, respectively. Two rules are used for fixing the decision variables:

(23) If $z_{LP} - h_1 z_{LP} < d_i$ then $x_i = 0$,

and

(24) if $z_{LP} - h_1 z_{LP} < -d_i$ then $x_i = 1$.

Where, d_i is the reduced cost for item i and h_1 ($h_1 < 1$) is a help variable whose value is chosen to be close to 1. Fixing these variables reduces the size of the search tree.

(b) It is well-known that the objective function value z_{LP} of the relaxed problem solution is an upper bound for the objective function value of the original 0-1 linear programming problem. Adema exploits this fact by initializing z_+ (i.e., the true lower bound of the optimal 0-1 objective function value) by $z_+ = h_2 z_{LP}$ ($0 << h_2 < 1$; $h_1 > h_2$.) instead of $z_+ = -\infty$. Then, the first

0–1 solution having an objective function value z between $h_2 z_{LP}$ and z_{LP} is accepted. This algorithm should not be used when z_{LP} is equal to zero, as no solution can be obtained. It is possible that no solution is found if h_1 or h_2 is too large, then, these values should be adapted.

Simulations

A simulation study was carried out in order to compare the four methods for parallel test construction described in the previous section. The computer program LANDO (Center for Mathematics and Computer Science) was used to solve the 0–1 programming problems; is based on the branch–and–bound algorithm developed by Land and Doig (1960). It was adapted according to the procedure of Adema (1988a) described previously, accepting 0–1 solutions that did not deviate more than 5% (h_2 = 0.95) from the relaxed objective function value (this value can be chosen smaller when the item bank is larger, and or when the item bank considered consists of items having small psychometric variations – e.g. when the Rasch model is considered). Furthermore, a value of h_1 = 0.999 for the help variable in (23) and (24) was used. The program was implemented on a DEC–2060 mainframe computer.

Two item banks each consisting of 100 items with the following properties were considered: 1) Rasch model, item difficulty parameters b~N(0,1), and 2) 3–parameter model, b~N(0,1), item discrimination parameters a~U(0.5,1.5), item

guessing parameters $c \sim U(0,0.25)$. In all simulations two parallel tests with diagnostic purposes were selected, for which the relative target information values r_k = 1, 1, 1 of the tests were specified at the ability levels θ = −1, 0, 1. Furthermore, the desired test length was N = 10 in all cases.

In this section, first, the influence of varying the maximum accepted difference p in test information values between two parallel tests on CPU-time (which includes reading the input and writing the output file) is examined for Method 3. Then, a comparison between the results from the various parallel test construction methods is made.

Both results of solving the relaxed and 0-1 problems are discussed below. The best look at the relaxed solutions is to consider them as upper bounds for the optimal solutions.

The Influence of Varying p for Method 3

The results from simulations varying the maximum accepted difference in test information values between the tests are summarized in Table 1. The values p = 0.5, 1.0, 2.0, 5.0 and ∞% were considered. Where ∞% stands for leaving out (12) and (13). Parallel tests were determined using Method 3. In the table the obtained objective function values z_{LP} for the relaxed problem and z_{0-1} for the 0-1 problem, the actual test information function values and the CPU-times needed are given. Furthermore, the relative differences of z_{LP} and z_{0-1} from z_{LP}, and the differences in test information between test 1 and 2 in percentages from the values of test 1 are summarized.

Insert Table 1 about here

Looking at the CPU-times for the relaxed problem it can be seen that there is not much variation. No clear trend towards lower CPU-times can be observed as the allowed differences in test information increase, only for p = ∞% CPU-times turn out to be significantly lower. CPU-times tend to be higher for the Rasch model. When the 0-1 solution is computed for the 3-parameter model, large CPU-times and large variations in CPU-times are noted. For the Rasch model the CPU-times and their variances are much smaller. When p = ∞% the CPU-times for the 0-1 problems are remarkable lower.

It is noted that mostly the differences in test information values for the Rasch model are much smaller than the maximum acceptable difference. In case of the 3-parameter model the differences are more in accordance with the maximum allowed percentage. When p = ∞% the differences are extremely small for the Rasch model; in the case of the 3-parameter model they are larger. This is to be expected because the shape of the item information function has more possibilities to vary in the 3-parameter model.

Comparing the Methods

In Tables 2 and 3 the results of simulations comparing the four methods are summarized for the Rasch and 3-parameter

model, respectively. For Methods 1 and 2 the second test was determined in two ways: without (t = 2a) and with (t = 2b) inclusion of constraints (6) and (7). In case of Methods 2 and 4, the item banks were partitioned into two subsets. This was done by ordering the items in the bank either on item difficulty (Ob) or on item discrimination (Oa), and subsequently randomly dividing the items from each next pair of items over the two different subsets. For all methods, excluding the construction of tests 2a for Methods 1 and 2, the maximum allowed difference between the actual test information values was p = 1%.

Insert Table 2 and 3 about here

Comparing the CPU-times needed to solve the relaxed problems (LP) in Tables 2 and 3, it can be seen that approximately equal times are required for the Rasch and 3-parameter model. However, Method 3 required a greater amount of time for the Rasch model (see also Table 1). Also, for the time needed to determine the 0-1 solutions, hardly any differences in CPU-time between the Rasch and 3-parameter model could be noted when Methods 1 and 2 (tests 2a) were used. However, for Methods 3 and 4 (see also Table 1) the Rasch model turned out to be faster. In general it was seen that Methods 3 and 4 were much slower than 1 and 2. Furthermore, Methods 2 and 4 (with partitions) turned out to

be quicker than their non-partitioned counterparts. For the 3-parameter model Method 2 had difficulties in determining tests 2b: When the item bank was ordered on difficulty it took a long time to determine the test, and in the case the item bank was ordered on discrimination no 0-1 solution was found at all.

Looking at the objective function values z_{LP} of the relaxed solutions for the Rasch model, it can be seen that they were all equal to 1.962 except for Method 1. For Methods 3 and 4, this implies that one of the tests had a lower bound equal to 1.962, whereas the other lower bound was slightly greater or equal to 1.962. Method 1 showed a large discrepancy between the z_{LP} values of both tests.

Next, the z_{LP} values for the 3-parameter model were compared. By summing the z_{LP} values of test 1 and 2b for Methods 1 and 2, and by taking two-times the z_{LP} values for Methods 3 and 4 these values could be compared. Thus, obtaining the values 4.588, 4.581, 4.866, 4.572 and 4.650 for Methods 2(Ob), 2(Oa), 3, 4(Ob) and 4(Oa), respectively. The largest objective function value was found for Method 3. The results for Methods 2 and 4 were slightly worse.

Comparing the results for the Rasch and the 3-parameter model, it is observed that the differences between z_{LP} and z_{0-1} were remarkably smaller for the Rasch model (< 1%, except for one case). The differences between tests were also much smaller for the Rasch model, even for tests 2a (< 1%, except for one test). For the 3-parameter model these differences were much larger; for Method 1 percentages of

about 25% were found (tests 2a). In two cases small changes in help variables h_1 and h_2 (used in (23) and (24)) were required for the 3-parameter model, because no solution could be obtained otherwise (see Table 3). For the 3-parameter model no solution was found for Method 1 tests 2b. Summing the results for the 3-parameter model, Method 1 determined tests that were far from parallel, Methods 3 and 4 gave very parallel tests, and Method 2 (tests 2b; item bank ordered on difficulty) performed slightly worse than 3 and 4. For the Rasch model approximately the same results in terms of differences in test information between the tests were obtained for all methods.

As for the items actually selected, it was seen that they had difficulty values close to 0 for the Rasch model. For the 3-parameter model there was a trend towards selecting items with discrimination values greater than 1, difficulty parameters varying between -1 and 1, and guessing parameters smaller than 0.12.

For Methods 1 and 3, the item banks were also ordered on difficulty and discrimination, respectively. The obtained results, which are not included in the tables, showed no improvement when ordered item banks were used.

Generalizability of the Results

In the above examples, only two item banks and one typical test were considered. However, the main point is that less CPU-time is required when less items with desired properties are available. For instance, the CPU-times would have been

lower if item banks with b ~ U[−3,3] were used, or if target values r_k = 4, 1, 1 for θ_k = −2, 0, 2 were specified. Also, including practical constraints would generally result in a decrease of CPU−time (Murtagh, 1981, Williams, 1978), since this reduces the solution space to be examined.

Discussion

In this paper four parallel test construction methods are discussed. The tests are selected from item banks on basis of their information functions using 0−1 linear programming. Methods 1 and 2 and Methods 3 and 4 select the tests sequentially and simultaneously, respectively. Methods 2 and 4 assume that the item banks are partitioned into subsets where the individual tests are constructed from.

It is concluded that when the Rasch model fits the items, sequential test construction methods are to be preferred. Both simultaneous and sequential methods are of equal accuracy; however, sequential methods require far less CPU−time. Care should be taken when Method 1 is used; it may give good results as long as there are enough items at hand. When there are no restrictions to the maximum allowed difference between the test information function values of the tests, simultaneous construction gives also accurate results in small amounts of time. For the 3−parameter model the use of Method 1 is absolutely inappropriate. In this case, it is recommended to use Method 2 with (6) and (7) or Method 4.

Only the psychometric aspects of the tests are considered in this paper. It is, however, possible to add all kinds of practical constraints, for instance, to control subject matter aspects. For a report on some of the possibilities the reader is referred to van der Linden and Boekkooi-Timminga (in press). Although in this reference the construction of one test at a time is considered, most constraints can easily be generalized to the simultaneous test construction problems in this paper.

References

Adema, J.J. (1988a). A note on solving large-scale zero-one programming problems (Research Report 88-4). Enschede, The Netherlands: University of Twente.

Adema, J.J. (1988b). The construction of two-stage tests (Research Report 88-14). Enschede, The Netherlands: University of Twente.

Birnbaum, A. (1968). Some latent trait models and their use in inferring an examinee's ability. In F.M. Lord & M.R. Novick, Statistical theories of mental test scores. Reading, Mass.: Addison-Wesley Publishing Company, Inc.

Boekkooi-Timminga, E. (1986). Algorithms for the construction of parallel tests by zero-one programming (Research Report 86-7). Enschede, The Netherlands: University of Twente.

Boekkooi-Timminga, E. (1987). Simultaneous test construction by zero-one programming. Methodika, 1, 101-112.

Boekkooi-Timminga, E. (1988). A cluster-based method for test construction. Submitted for publication.

Center for Mathematics and Computer Science. LANDO. The Netherlands: Amsterdam.

Dantzig, G. (1957). Discrete-variable extremum problems. Operations Research, 5, 266-277.

Gulliksen, H. (1950). Theory of mental tests. New York: John Wiley & Sons, Inc.

Kelderman, H. (1987). Some procedures to assess target information functions. In W.J. van der Linden (Ed.), IRT-based test construction (Research Report 87-2). Enschede, The Netherlands: University of Twente.

Land, A.H., & Doig, A. (1960). An automatic method of solving discrete programming problems. Econometrica, 28, 497-520.

Lenstra, J.K., & Rinnooy Kan, A.H.G. (1979). Computational complexity of discrete optimization problems. In P.L. Hammer, E.L. Johnson, & B.H. Korte (Eds.), Discrete optimization I. New York: North-Holland Publishing Company.

Murtagh, B.A. (1981). Advanced linear programming: Computation and practice. New York: McGraw-Hill.

O'hEigeartaigh, M., Lenstra, J.K., & Rinnooy Kan, A.H.G. (Eds.). (1985). Combinatorial optimization: Annotated bibliographies. New York: John Wiley & Sons.

Rao, S.S. (1985). Optimization: Theory and applications (2nd Ed.). New Delhi: Wiley Eastern Ltd.

Salkin, H.M. (1975). Integer programming. London: Addison-Wesley Publishing Company.

Samejima, F. (1977). Weakly parallel tests in latent trait theory with some criticisms of classical test theory. Psychometrika, 42, 193-198.

Taha, H.A. (1975). Integer programming. New York: Academic Press.

Theunissen, T.J.J.M. (1985). Binary programming and test design. Psychometrika, 50, 411-420.

Theunissen, T.J.J.M. (1986). Optimization algorithms in test design. Applied Psychological Measurement, 10, 381–390.

Timminga, E. (1985). Geautomatiseerd toetsontwerp: Itemselectie m.b.v. binair programmeren [Automated test design: Item selection by binary programming]. Unpublished master's thesis, University of Twente, Enschede, The Netherlands.

van der Linden, W.J., & Boekkooi–Timminga, E. (1988). A zero–one programming approach to Gulliksen's matched random subtests method. Applied Psychological Measurement, 12, 201–209.

van der Linden, W.J., & Boekkooi–Timminga, E. (in press). A maximin model for test design with practical constraints. Psychometrika.

Wagner, H.M. (1972). Principles of operations research: with applications to managerial decisions. London: Prentice–Hall International, Inc.

Williams, H.P. (1978). Model building in mathematical programming. New York: John Wiley.

Table 1
Varying the Maximum Accepted Difference in Percentages between Test Information Values of the Tests for Method 3 Model 2

t	%	z_{LP}	z_{0-1}	Test Information I(-1)	Test Information I(0)	Test Information I(1)	CPU-time LP	CPU-time 0-1
Rasch model								
1	0.5	1.962	1.931 (1.6%)	1.978	2.473	1.939	43.0	82.9
2				1.985 (0.4%)	2.471 (0.1%)	1.931 (0.4%)		
1	1.0	1.962	1.947 (0.8%)	1.961	2.480	1.960	42.6	79.9
2				1.947 (0.7%)	2.470 (0.4%)	1.967 (0.4%)		
1	2.0	1.962	1.950 (0.6%)	1.950	2.456	1.955	32.8	52.3
2				1.956 (0.3%)	2.488 (1.3%)	1.969 (0.7%)		
1	5.0	1.962	1.951 (0.6%)	1.966	2.474	1.951	37.5	94.7
2				1.960 (0.3%)	2.474 (0.0%)	1.958 (0.4%)		
1	-	1.962	1.959 (0.2%)	1.959	2.484	1.964	16.8	32.7
2				1.959 (0.0%)	2.485 (0.0%)	1.964 (0.0%)		
3-parameter model								
1	0.5	2.433	2.333 (4.1%)	2.420	2.996	2.333	22.1	89.3
2				2.418 (0.1%)	3.007 (0.4%)	2.339 (0.3%)		
1	1.0	2.433	2.332 (4.2%)	2.408	2.999	2.332	17.9	146.4
2				2.430 (0.9%)	3.004 (0.2%)	2.340 (0.3%)		
1	2.0	2.433	2.340 (3.8%)	2.340	3.123	2.493	21.3	372.6
2				2.376 (1.5%)	3.103 (0.6%)	2.491 (0.1%)		
1	5.0	2.433	2.359 (3.0%)	2.403	2.920	2.359	18.6	59.1
2				2.411 (0.3%)	3.032 (3.5%)	2.366 (0.3%)		
1	-	2.433	2.328 (4.3%)	2.453	2.959	2.349	14.1	29.9
2				2.328 (5.1%)	3.048 (3.0%)	2.420 (3.0%)		

Table 2

Results for the Methods when the Rasch Model fits

Method				Test Information			CPU-time	
	t	z_{LP}	z_{0-1}	I(-1)	I(0)	I(1)	LP	0-1
1	1	1.964	1.960 (0.2%)	1.960	2.490	1.967	2.5	3.4
	2a	1.961	1.958 (0.2%)	1.958 (0.1%)	2.479 (0.4%)	1.963 (0.2%)	2.4	3.4
	2b	1.961	1.951 (0.5%)	1.951 (0.5%)	2.476 (0.6%)	1.967 (0.0%)	2.5	4.0
2 Ob	1	1.962	1.954 (0.4%)	1.964	2.477	1.954	1.6	2.1
	2a	1.962	1.925 (1.9%)	1.996 (1.6%)	2.479 (0.1%)	1.925 (1.5%)	1.6	2.2
	2b	1.962	1.960 (0.1%)	1.960 (0.2%)	2.481 (0.2%)	1.961 (0.4%)	1.4	2.3
3	1	1.962	1.947 (0.8%)	1.961	2.480	1.960	42.6	79.9
	2			1.947 (0.7%)	2.470 (0.4%)	1.967 (0.4%)		
4 Ob	1	1.962	1.942 (1.0%)	1.942	2.473	1.975	10.4	18.4
	2			1.958 (0.8%)	2.468 (0.2%)	1.956 (1.0%)		

Ob: item bank ordered on item difficulty

Table 3

Results for the Methods when the 3-Parameter Model fits

Method		t	z_{LP}	z_{0-1}	Test Information I(-1)	I(0)	I(1)	CPU-time LP	CPU-time 0-1
1		1	2.681	2.615 (2.5%)	2.746	3.458	2.615	2.2	2.4
		2a●	2.230	2.092 (6.2%)	2.092 (23.8%)	2.631 (23.9%)	2.211 (15.5%)	2.4	3.9
		2b	not feasible						
2	Ob	1	2.286	2.279 (0.3%)	2.279	3.085	2.294	1.2	1.6
		2a	2.530	2.520 (0.4%)	2.520 (10.6%)	3.194 (3.5%)	2.546 (8.6%)	1.3	1.4
		2b	2.302	2.275 (1.2%)	2.275 (0.2%)	3.089 (0.1%)	2.305 (0.5%)	1.3	11.6
	Oa	1	2.325	2.234 (3.9%)	2.234	3.279	2.428	1.4	1.6
		2a	2.493	2.456 (1.5%)	2.527 (13.1%)	3.093 (5.7%)	2.456 (1.2%)	1.3	1.4
		2b	2.256	no 0-1 solution				1.3	1.8
3		1	2.433	2.332 (4.2%)	2.408	2.999	2.332	17.9	146.4
		2			2.430 (0.9%)	3.004 (0.2%)	2.340 (0.3%)		
4	Ob	1	2.286	2.279 (0.3%)	2.279	3.085	2.294	8.8	30.8
		2			2.298 (0.8%)	3.070 (0.5%)	2.279 (0.7%)		
	Oa	1●	2.325	2.208 (5.0%)	2.208	3.019	2.442	8.1	106.4
		2●			2.228 (0.9%)	3.023 (0.1%)	2.457 (0.6%)		

●: $h_1 = 0.99$; $h_2 = 0.93$

Oa: item bank ordered on item discrimination values

Ob: item bank ordered on item difficulty values

Author's Note

The Netherlands Organization for Scientific Research (N.W.O.) is gratefully acknowledged for funding this project. This research was conducted while Ellen Boekkooi-Timminga was supported by a PSYCHON-grant of this organization (560-267-001), awarded to Dr. W.J. van der Linden.

I would like to thank Tom A.B. Snijders, University of Groningen, The Netherlands for his remarks on an earlier version of Method 3. Furthermore, I would like to thank Jos J. Adema and Wim J. van der Linden, University of Twente, The Netherlands for their assistance during the preparation of this paper.

Titles of recent Research Reports from the Division of Educational Measurement and Data Analysis, University of Twente, Enschede, The Netherlands.

RR-88-1 E. van der Burg & J. de Leeuw, *Nonlinear redundancy analysis*

RR-88-2 W.J. van der Linden & J.J. Adema, *Algorithmic test design using classical item parameters*

RR-88-3 E. Boekkooi-Timminga, *A cluster-based method for test construction*

RR-88-4 J.J. Adema, *A note on solving large-scale zero-one programming problems*

RR-88-5 W.J. van der Linden, *Optimizing incomplete sample designs for item response model parameters*

RR-88-6 H.J. Vos, *The use of decision theory in the Minnesota Adaptive Instructional System*

RR-88-7 J.H.A.N. Rikers, *Towards an authoring system for item construction*

RR-88-8 R.J.H. Engelen, W.J. van der Linden, & S.J. Oosterloo, *Item information in the Rasch model*

RR-88-9 W.J. van der Linden & T.J.H.M. Eggen, *The Rasch model as a model for paired comparisons with an individual tie parameter*

RR-88-10 H. Kelderman & G. Macready, *Loglinear-latent-class models for detecting item bias*

RR-88-11 D.L. Knol & M.P.F. Berger, *Empirical comparison between factor analysis and item response models*

RR-88-12 E. van der Burg & G. Dijksterhuis, *Nonlinear canonical correlation analysis of multiway data*

RR-88-13 J. Kogut, *Asymptotic distribution of an IRT person fit index*

RR-88-14 J.J. Adema, *The construction of two-stage tests*

RR-88-15 H.J. Vos, *Simultaneous optimization of decisions using a linear utility function*

RR-88-16 H. Kelderman, *An IRT model for item responses that are subject to omission and/or intrusion errors*

RR-88-17 H. Kelderman, *Loglinear multidimensional IRT models for polytomously scored items*

RR-88-18 H.J. Vos, *Applications of decision theory to computer based adaptive instructional systems*

RR-89-1 R.J.H. Engelen & R.J. Jannarone, *A connection between item/subtest regression and the Rasch model*

RR-89-2 E. Boekkooi-Timminga, *The construction of parallel tests from IRT-based item banks*

Research Reports can be obtained at costs from Bibliotheek, Department of Education, University of Twente, P.O. Box 217, 7500 AE Enschede, The Netherlands.

department of
EDUCATION

A publication by
the Department of Education
of the University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands